Graph Matching for Corpora Exploration

Bruno Guillaume
Université de Lorraine, CNRS, Inria, LORIA, F-54000 Nancy, France

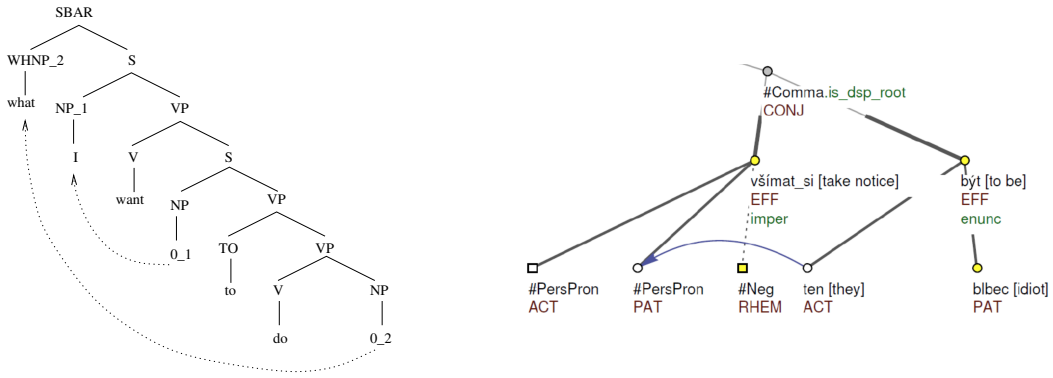
Abstract

The goal of this paper is to advocate for the usage of graph structures for linguistic representations and of graph methods for corpus linguistics. We describe our graph-based tool (GREW-MATCH) and show a few examples of applications both on syntactically and semantically annotated corpora.

1 Introduction

When annotated corpora are available, it is possible to conduct more fined-grained corpus linguistics studies. In this paper, we will focus on two kinds of corpora: syntactically and semantically annotated corpora.

When dealing with syntax, most linguistic models are based on tree structures. The two mainstream syntactic approaches: Phrase Structure and Dependencies propose to represent the syntax of a sentence as a tree. Nevertheless, if we have a closer look to popular corpora, they introduced mechanisms that are going beyond tree structure. The figure below on the left is a subpart of the first example given in Stephen Clark's presentation *Penn Treebank Parsing*[1] [TMS03]. There are additional (dotted) edges which describe the links between some traces and their antecedent. If we consider that these links are in the structure, it is no longer a tree but a graph. Similarly, the figure below on the right is the first example given in the documentation page of the Prague Dependency Treebank[2] [HHMM17]. The blue arrow links two nodes with a common ancestor and so the structure is a graph.



When semantics is taken into account, most formalisms rely either an explicit notion of graphs or on other mathematical structures (logical formulae or λ-terms) that can be represented as graphs.

These examples drive us to consider all these structures as graphs and to propose GREW-MATCH, a graph-based tools to deal with.

2 Graph Matching

We consider an usual mathematical definition of a *graph*: a set of *nodes* and a set of relations between the nodes, called *edges*). To cope with Natural Language Processing specificities, we add feature structures on nodes (to handle phonological forms, lemmas, POS, morphological features...) and labels on edges to be able to type relations.

[1] http://www.cs.ox.ac.uk/files/552/stat_parse2.pdf
[2] https://ufal.mff.cuni.cz/pdt3.0/documentation

Graph matching is also a mathematical notion which is well defined: it is the process of searching for a *pattern* (which is expressed itself as a graph) and finding all possible ways to recognize the pattern in the host graph. We talk about an *occurrence* of the pattern for each solution of the search.

In the tool, patterns are written in a dedicated syntax with three kind of constraints:

- Global constraints about the whole graph (testing for the presence of cycles or for a tree structure),
- Positive constraints where the user describes a set of nodes, of relations and constraints on them,
- Negative constrainsts which are used as filters on the positive constraint output.

All these part are optional and negative constraints can be repeated. If there are more than one negative constraints, they are intrepreted as successive independant filters. Some examples of each constraint are given below.

For a convenient usage, an online web application is available[3]. The user selects a corpus (a few hundreds are proposed) and writes a graph pattern. The tool returns the numbers of occurrences found in the corpus and displays some of the results. A tutorial mode is available to help new users to learn the concrete syntax of patterns.

3 Application to syntactically annotated corpora

The tool is available on all Universal Dependencies [NdMG+16] (UD) corpora and the examples below are done on one of them: the UD_English-GUM [Zel17] which contains 4,399 annoted trees and 80 176 tokens (in UD version 2.3).

Our first example computes some statistics about the `conj` relation which is used in UD to link heads of conjucts in a coordination construction. On the UD_English-GUM corpus, we observe 2,563 occurrences of `conj` relations with two homogenoeous conjuncts (first pattern below) and 535 occurrences of coordination of unlikes (second pattern). Among the 535, the most productive pair of POS is when `C1` is adjective whereas `C2` is a verb (third pattern) with 71 occurences. An example of the last case is *They are simply afraid and hate modernity.* where the `conj` relation links *afraid* and *hate*.

```
pattern { C1 -[conj]-> C2 ; C1.upos = C2.upos }
pattern { C1 -[conj]-> C2 ; C1.upos <> C2.upos }
pattern { C1 -[conj]-> C2 ; C1[upos=ADJ]; C2[upos=VERB] }
```

With other patterns, we can examine the most frequent POS of conjucts in the 2,563 occurrences: there are nouns (1,102), verbs (868), proper names (396), adjectifves (187).

Our second example explores different kinds of noun phrases (NP). We would like to estimate the proportion of NP built with or without a determiner and to see if these proportion varies depending of the syntactic role of the NP in the sentence. We obseve that a majority (59.3%) of the nouns of the corpus are used without any determiner. The table below gives these proportion for some syntactic role of nouns.

	subj	obj	obl	nmod	compound	other	Total
with	1,070	1,412	1,470	1,047	8	1,043	6,050
det	50.9%	48.7%	50.4%	43.3%	0.5%	35.2%	40.7%
without	1,031	1,488	1,447	1,371	1,554	1,922	8,813
det	49.1%	51.3%	49.6%	56.7%	99.5%	64.8%	59.3%
Total	2,101	2,990	2,917	2,418	1,562	2,965	14,863

Patterns with negative constrains are used to get some numbers of the table. For instance, the 1,488 occurrences of direct object nouns without determiner are found with the pattern:

```
pattern { N[upos=NOUN]; * -[obj]-> N;}
without { N -[det]-> *; }
```

[3] http://match.grew.fr

4 Application to semantically annotated corpora

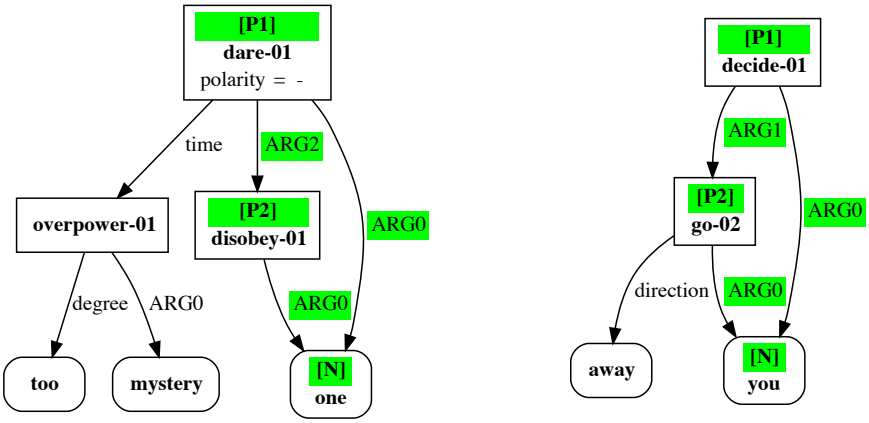
We have chosen to use the Abstract Meaning Representation [BBC+13] (AMR) because it provides freely available annotated corpora but the method may be applied to other semantic framework as soon as we are able to convert the annotatated structures into graphs that can be read by the tool.

From the AMR website[4], two corpora are available: the translation in English of the Saint-Exupéry's novel *The Little Prince* (1,562 sentences) and the Bio AMR Corpus (6,452 sentences from 3 full cancer-related PubMed articles). Both corpora are available on the online tool but we will here focus on *The Little Prince* where sentences are much shorter and simpler. In this section, in order to spot the potential usages of the tool, we focus only on examples where the graph structure is essential.

In AMR, relations between predicates and arguments are encoded with labels `ARG0` (roughly for the semantic agent) and `ARG1`, `ARG2`, ..., `ARG5` for other predicative roles. The following pattern describes constructions where two predicates `P1` and `P2` share the same `ARG0` argument and such that `P2` is an argument of `P1`. Note that it is not possible to search for this kind of structure if the pattern itself cannot have a graph structure (because if the sharing of `N`).

```
pattern { P1 -[ARG1|ARG2|ARG3]-> P2; P1 -[ARG0]-> N; P2 -[ARG0]-> N; }
```

141 occurrences of the pattern are found in the corpus. Two of them are shown in the figure below (these figures are also an example of the graphical output avalaible in the online tool where parts of the graph corresponding to the pattern image are highlighted).



One can also obtain some statistics about the occurrences, for instance about the predicates involved in this construction. In the previous example, around 50 different predicates are used in the `P1` node for this construction. The most frequent predicate is by far `say-01` (23 occurrences); the next ones being `begin-01` (7 occurrences), `continue-01`, `try-01`, `want-01` (each with 6 occurrences).

As say earlier, the tool allows also to search for global properties on the graphs. In AMR guidelines[5], it said that: "*Approximately 0.3% of AMRs are legitimately cyclic*"; but we can observe that it is underestimated at least for the two available corpora. With the pattern `global { is_cyclic }`, we found 35 sentences with AMR containing a cycle (2.24% of the 1,563 sentences). With patterns on nodes; we can explore further these cycles (6 are of length 2; 26 of length 3 and 8 of length 4). In the Bio AMR Corpus, 2.71% of the graphs are cyclic.

5 Conclusion

The tool we propose can be used in many different ways to explore the graph structure of annotated linguistic corpora. Is is currently used on syntax and on semantics but usages on other kinds of corpora can be imagined. In fact, the method can be used on any graph structure and for instance, it has been recently used on lexical databases.

[4] https://amr.isi.edu
[5] https://github.com/amrisi/amr-guidelines/blob/master/amr.md#cycles

References

[BBC+13] Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. Abstract meaning representation for sembanking. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, pages 178–186, 2013.

[HHMM17] Jan Hajič, Eva Hajičová, Marie Mikulová, and Jiří Mírovskỳ. Prague dependency treebank. In *Handbook of Linguistic Annotation*, pages 555–594. Springer, 2017.

[NdMG+16] Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Yoav Goldberg, Jan Hajic, Christopher D Manning, Ryan McDonald, Slav Petrov, Sampo Pyysalo, Natalia Silveira, et al. Universal dependencies v1: A multilingual treebank collection. In *Proceedings of LREC 2016*, pages 1659–1666, 2016.

[TMS03] Ann Taylor, Mitchell Marcus, and Beatrice Santorini. *The Penn Treebank: An Overview*, pages 5–22. Springer Netherlands, Dordrecht, 2003.

[Zel17] Amir Zeldes. The GUM corpus: Creating multilayer resources in the classroom. *Language Resources and Evaluation*, 51(3):581–612, 2017.